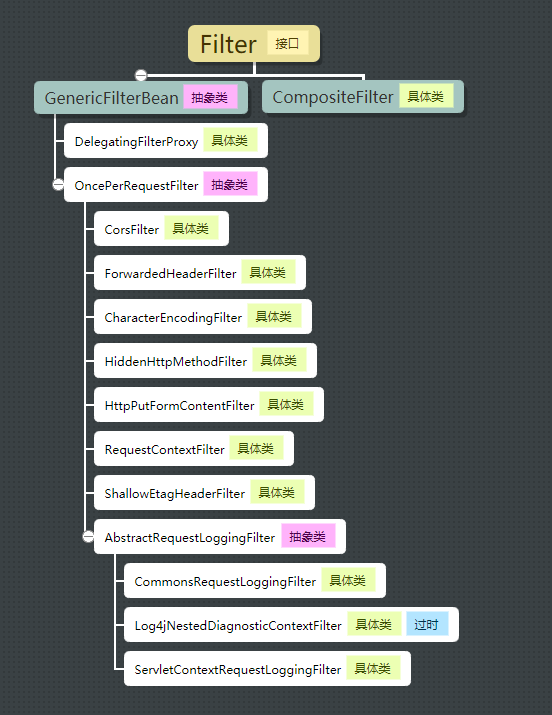
# 过滤器详解



背景：Springweb包中中有很多过滤器，这些过滤器位于org.springframework.web.filter并且理所当然地实现了javax.servlet.Filter。

# 超类（抽象类）GenericFilterBean

## 应用背景

### 作用

1. 实现了Filter的抽象类，作为一个超类供其他类继承，其中的init和destory在该类中实现，doFilter在具体实现类中实现。
2. GenericFilterBean中包含一个内部私有类FilterConfigPropertyValues，主要用于将web.xml中定义的init-param的值取出。
3. 该类的主要作用是：从web.xml文件中取得init-param中设定的值，然后对Filter进行初始化（当然，其子类可以覆盖init方法）。

### 实现的接口

1. **Filter**，实现过滤器；
2. **BeanNameAware**，实现该接口的setBeanName方法，便于Bean管理器生成Bean；
3. **EnvironmentAware**，实现该接口的setEnvironment方法，指明该Bean运行的环境；
4. **ServletContextAware**，实现该接口的setServletContextAware方法，指明上下文；
5. **InitializingBean**，实现该接口的afterPropertiesSet方法，指明设置属性生的操作；
6. **DisposableBean**，实现该接口的destroy方法，用于回收资源。

### 主要方法

1. 已实现方法：init(),destroy()
2. 子类方法：doFilter()

# 抽象类OncePerRequestFilter

## 应用背景

### 作用

1. OncePerRequestFilter继承于GenericFilterBean，保留了GenericFilterBean中的所有方法并对之进行了扩展，在oncePerRequestFilter中的主要方法是doFilter。
2. OncePerRequestFilter主要作用是实现了doFilter方法，只支持HTTP请求，确保在接收到一个request后，每个filter只执行一次，它的子类只需要关注Filter的具体实现即doFilterInternal。
3. 在doFilter方法中，doFilterInternal方法由子类实现，主要作用是规定过滤的具体方法。

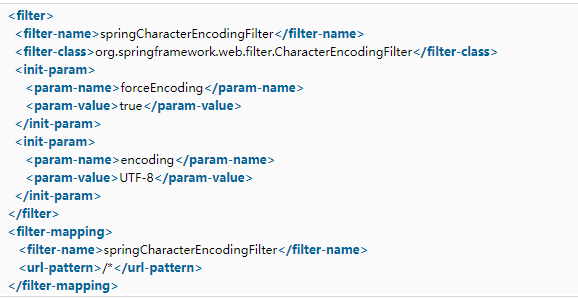
### 主要方法

1. 已实现方法：dofilter()
2. 子类实现方法：doFilterInternal()

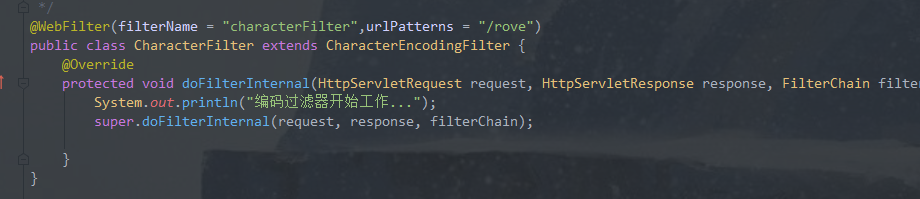
## 字节码过滤器CharacterEncodingFilter

### 使用过滤器

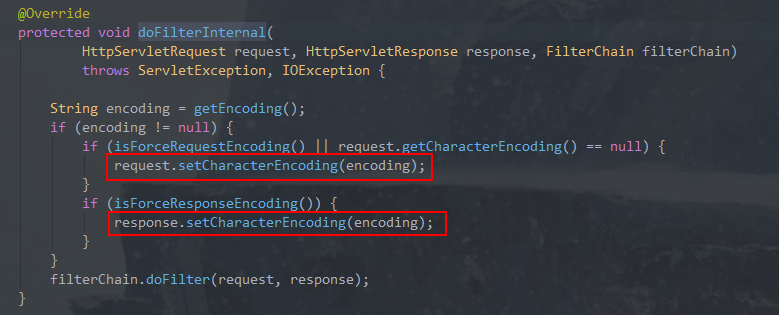
(1)在配置文件中设置过滤器



(2)利用注解@WebFilter



### 编码过滤器原理

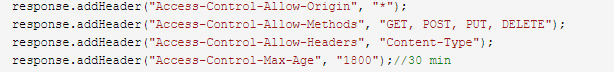


1. request.setCharacterEncoding()只对POST请求有效，对GET请求无效。

## 跨域过滤器CorsFilter

### 跨域的解决方法

1. 设置返回头



1. 跨域过滤器

CorsFilter过滤器，目前不会使用。

1. 注解@CrossOrigin,用在类上或者方法上



1. 利用springWeb自带的配置接口WebMvcConfigurer



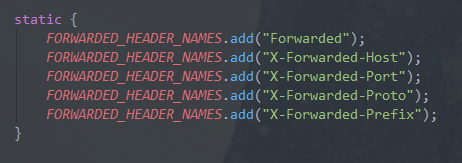
1. support是一个mvc自带的快速配置类。另一种方法是注解@EnableWebMvc可以结合WebMvcConfigurer接口使用，不用实现所有类。

## 重定向头过滤器ForwardedHeaderFilter

### 应用背景

1. 由于请求会经过像负载均衡器这样的代理，那么host，port，scheme在创建一些资源文件的链接的时候返回给客户端可能是有所不同的。
2. 该过滤器会将非标准的HTTP头部，转换为标准的 HTTP头部。

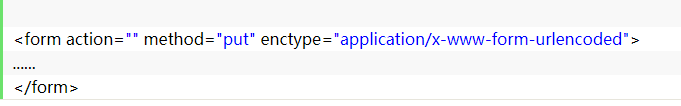
（非标准头部）



## PUT请求表单HttpPutFormContentFilter

### 应用背景

1. 浏览器只能通过GET或者POST提交FORM数据，但是非浏览器的客户端可以使用PUT或者PATCH。
2. Servlet API仅为POST方法提供了ServletRequest.getParameter\*()方法获取FORM信息。
3. spring-web模块提供了HttpPutFormContentFilter检查PUT或者PATCH方法的**enctype是否是application/x-www-form-urlencoded,**如果是，则从请求体重读取属性并封装到ServletRequest中，以便日后通过ServletRequest.getParameter\*()获取FORM数据。(对请求头enctype有严格的要求)



1. 总的来说，就是该过滤器可以获取到PUT请求的表单数据，提取到request中，而后可以通过getParameter方法去获取。

## 请求转换过滤器HiddenHttpMethodFilter

### 应用背景

1. 过滤器主要作用：将POST请求转换为DELETE(删除)或PUT(更新)请求。
2. 实现的原理就是对请求进行过滤，源码中实现是拦截POS请求，获取隐藏域\_method的值，将POST替换为对应的\_method的value值，从而实现POST转换为DELETE或PUT

### 过滤器使用

1. 通过设置一个隐藏域的值，将POST请求转换为DELETE或者PUT请求。

（前端表单）

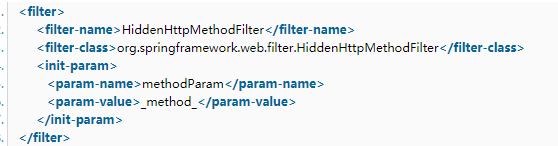
<form action="test11" method="POST">

<input type="hidden" name="\_method" value="DELETE"/>

<input type="submit" value="DELETE"/>

</form>

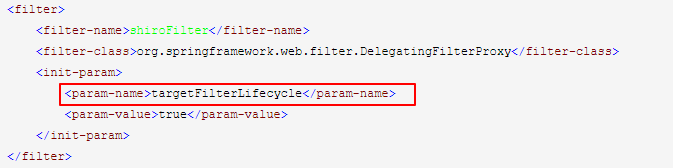
1. 自定义隐藏域的名称(改动\_method)



## 代理过滤器DelegatingFilterProxy

### 应用背景

1. 作用就是一个filter的代理，用这个类的好处是可以通过spring容器来管理filter的生命周期，还有就是，可以通过spring注入的形式，来代理一个filter执行，如shiro。
2. 配置界面



标签<param-name>就是需要被代理的类。

## 请求分发过滤器RequestContextFilter

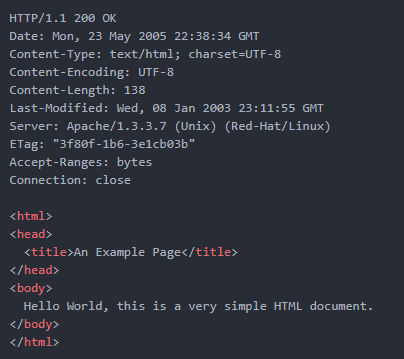
### 应用背景

1. spring中有request请求转发的功能，将页面提交的request转发到Controller中。
2. 在web.xml中配置RequestContextListener和DispatcherServlet，利用中央分发器进行转发。
3. 在web.xml中配置RequestContextFilter，这种方式通常是用于配置第三方servlet，就是没有springMVC的DispatcherServlet的时候，就是通过过滤器RequestContext进行转发。

## 生成Etag过滤器ShallowEtagHeaderFilter

### Etag简介

1. ETag 是 Entity Tag 的缩写，中文译过来就是实体标签的意思。在HTTP1.1协议中其实就是请求HEAD中的一个属性而已。



1. ETag是HTTP1.1中才加入的一个属性，**用来帮助服务器控制Web端的缓存验证**。它的原理是这样的，当浏览器请求服务器的某项资源(A)时, 服务器根据A算出一个哈希值(3f80f-1b6-3e1cb03b)并通过 ETag 返回给浏览器，浏览器把"3f80f-1b6-3e1cb03b" 和 A 同时缓存在本地，当下次再次向服务器请求A时，会通过类似 If-None-Match: "3f80f-1b6-3e1cb03b" 的请求头把ETag发送给服务器，服务器再次计算A的哈希值并和浏览器返回的值做比较，如果发现A发生了变化就把A返回给浏览器(200)，如果发现A没有变化就给浏览器返回一个304未修改。这样通过控制浏览器端的缓存，可以节省服务器的带宽，因为服务器不需要每次都把全量数据返回给客户端。

### 过滤器作用

1. **ShallowEtagHeaderFilter会将JSP等的内容缓存，生成MD5的key，然后在response中作为Etage的header返回给客户端。**下次客户端对相同的资源（或者说相同的url）发出请求时，客户端会将之前生成的key作为If-None-Match的值发送到server端。 **Filter会客户端传来的值和服务器上的做比较**，如果相同，则返回304；否则，将发送新的内容到客户端。

# 抽象类AbstractRequestLoggingFilter

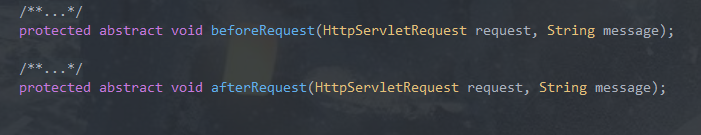
## 应用背景

### 作用(上下文信息过滤器)

1. AbstractRequestLoggingFilter是对OncePerRequestFilter的扩展，它除了遗传了其父类及祖先类的所有功能外，还在doFilterInternal中决定了在过滤之前和之后执行的事件，它的子类关注的是beforeRequest和afterRequest。
   1. 主要方法
2. 已实现方法doFilterInternal



1. 子类关注方法



## 各个具体过滤器作用

1. CommonsRequestLoggingFilter在过滤前后分别打印出一段debug的信息.
2. ServletContextRequestLoggingFilter在过滤前后分别向日志文件中写入一段日志信息，日志文件可由log4j.properties等指定
3. Log4jNestedDiagnosticContextFilter则将日志信息存储到NDC中，NDC采用了一个类似栈的机制来push和pot上下文信息，每一个线程都独立地储存上下文信息，比如说一个servlet就可以针对 每一个request创建对应的NDC，储存客户端地址等信息。
4. 在spring中已定义的AbstractRequestLoggingFiter的子类过滤器均是用于输入上下文信息，只不过承载的方式不同，其中CommonsRequestLoggingFilter将上下文信息直接打印；ServletContextRequestLoggingFilter将上下文信息写入日志文件；Log4jNestedDiagnosticContextFilter将上下文信息写入NDC中。